NAME

t_connect – establish a connection with another transport user

SYNOPSIS

cc [*flag* ...] *file* ... **–lnsl** [*library* ...]

#include <tiuser.h>

int t_connect(int *fildes***, const struct t_call** ∗*sndcall*, **struct t_call** ∗*rcvcall*);

MT-LEVEL

MT-Safe

DESCRIPTION

This function enables a transport user to request a connection to the specified destination transport user. *fildes* identifies the local transport endpoint where communication will be established, while *sndcall* and *rcvcall* point to a **t_call** structure that contains the following members:

```
struct netbuf    addr;
struct netbuf    opt;
struct netbuf    udata;
int              sequence;
```

sndcall specifies information needed by the transport provider to establish a connection and *rcvcall* specifies information that is associated with the newly established connection.

The address is specified in the **netbuf** structure which has the following format:

```
struct netbuf {
        unsigned int maxlen;
        unsigned int len;
        char *buf;
}
```

where **maxlen** specifies the maximum length of the buffer in bytes, **len** specifies the bytes of data in the buffer, and **buf** points to the buffer that contains the data.

In *sndcall*, **addr** specifies the protocol address of the destination transport user, **opt** presents any protocol-specific information that might be needed by the transport provider, **udata** points to optional user data that may be passed to the destination transport user during connection establishment, and **sequence** has no meaning for this function.

On return in *rcvcall*, **addr** returns the protocol address associated with the responding transport endpoint, **opt** presents any protocol-specific information associated with the connection, **udata** points to optional user data that may be returned by the destination transport user during connection establishment, and **sequence** has no meaning for this function.

The **opt** argument implies no structure on the options that may be passed to the transport provider. The transport provider is free to specify the structure of any options passed to it. These options are specific to the underlying protocol of the transport provider. The user may choose not to negotiate protocol options by setting the **len** field of **opt** to zero. In this case, the provider may use default options.

The **udata** argument enables the caller to pass user data to the destination transport user and receive user data from the destination user during connection establishment. However, the amount of user data must not exceed the limits supported by the transport provider as returned in the **connect** field of the **info** argument of **t_open**(3N) or **t_getinfo**(3N). If the **len** (see **netbuf** in **t_connect**(3N)) field of **udata** is zero in *sndcall*, no data will be sent to the destination transport user.

On return, the **addr**, **opt**, and **udata** fields of *rcvcall* will be updated to reflect values associated with the connection. Thus, the **maxlen** (see **netbuf** in **t_connect**(3N)) field of each argument must be set before issuing this function to indicate the maximum size of the buffer for each. However, *rcvcall* may be NULL, in which case no information is given to the user on return from **t_connect( )**.

By default, **t_connect( )** executes in synchronous mode, and will wait for the destination user's response before returning control to the local user. A successful return (that is, return value of zero) indicates that the requested connection has been established. However, if **O_NDELAY** or **O_NONBLOCK** is set (using **t_open**(3N) or **fcntl**(2)**), t_connect( )** executes in asynchronous mode. In this case, the call will not wait for the remote user's response, but will return control immediately to the local user and return -1 with **t_errno** set to **TNODATA** to indicate that the connection has not yet been established. In this way, the function simply initiates the connection establishment procedure by sending a connect request to the destination transport user.

RETURN VALUES

t_connect( ) returns 0 on success. On failure **t_connect( )** returns –1, **t_errno** is set to indicate the error, and possibly **errno** is set.

ERRORS

On failure, **t_errno** will be set to one of the following:

TACCES	The user does not have permission to use the specified address or options.
TBADADDR	The specified protocol address was in an incorrect format or contained illegal information.
TBADDATA	The amount of user data specified was not within the bounds allowed by the transport provider.
TBADF	The specified file descriptor does not refer to a transport endpoint.
TBADOPT	The specified protocol options were in an incorrect format or contained illegal information.
TBUFOVFLW	The number of bytes allocated for an incoming argument is not sufficient to store the value of that argument. If executed in synchronous mode, the provider's state, as seen by the user, changes to **T_DATAXFER**, and the connect indication information to be returned in *rcvcall* is discarded.
TLOOK	An asynchronous event has occurred on this transport endpoint and requires immediate attention.
TNODATA	**O_NDELAY** or **O_NONBLOCK** was set, so the function successfully initiated the connection establishment procedure, but did not wait

for a response from the remote user.

TNOTSUPPORT This function is not supported by the underlying transport provider.

TOUTSTATE The function was issued in the wrong sequence.

TSYSERR A system error has occurred during execution of this function, **errno** will be set to the specific error.

SEE ALSO

fcntl(2), **t_accept**(3N), **t_getinfo**(3N), **t_listen**(3N), **t_open**(3N), **t_optmgmt**(3N), **t_rcvconnect**(3N)

Transport Interfaces Programming Guide

NOTES

This interface is safe in multithreaded applications.